



20.02.24 09:57

CSCI 165

Introduction to the Internet and the World Wide Web

Lecture 5: Javascript 2



Jetic Gū
2024 Spring Semester (S1)

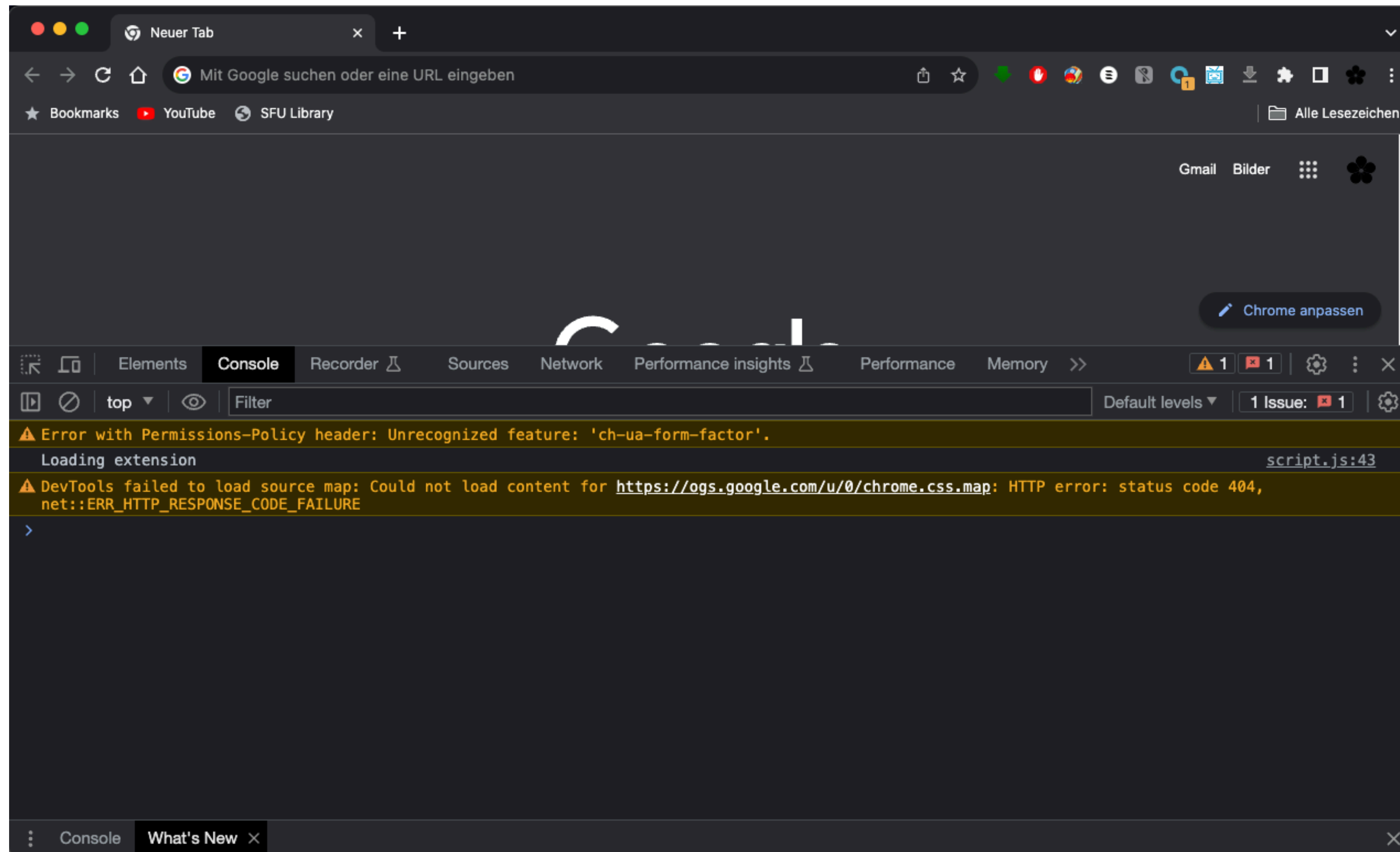
Overview

- Focus: Course Introduction
- Architecture: WWW
- Core Ideas:
 1. Numerical Calculations, Variables, Function
 2. Data Types

Calculation in Javascript

- Javascript can carry out some basic calculation
- First, let's take a look at a console
 - Every modern browser allows you to access the console in the developer tool

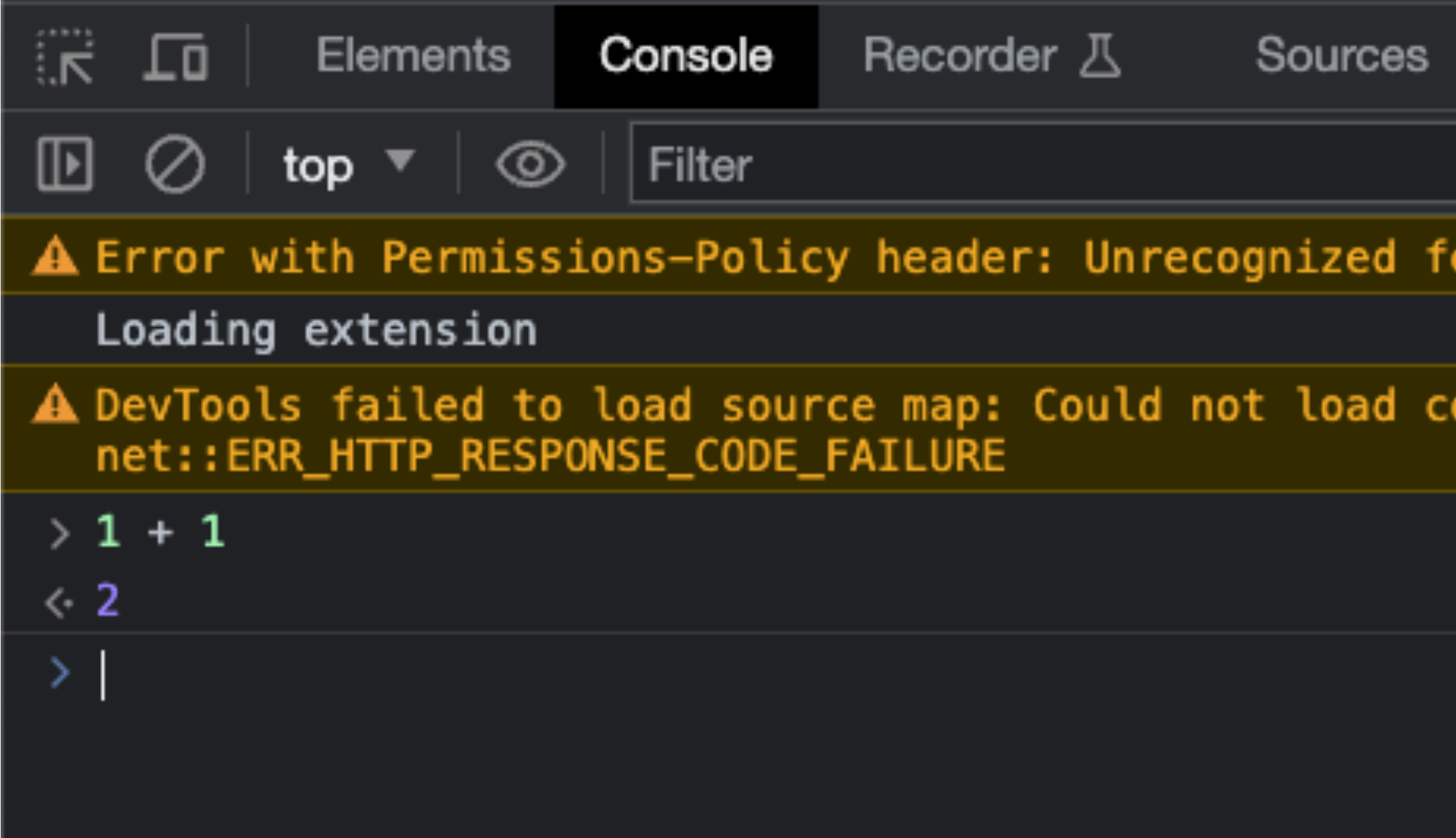
Calculation in Javascript



Technical

Calculation in Javascript

- You can execute Javascript function calls, as well as general statements etc. here.
- Try some basic calculation of numbers. These are called Expressions¹



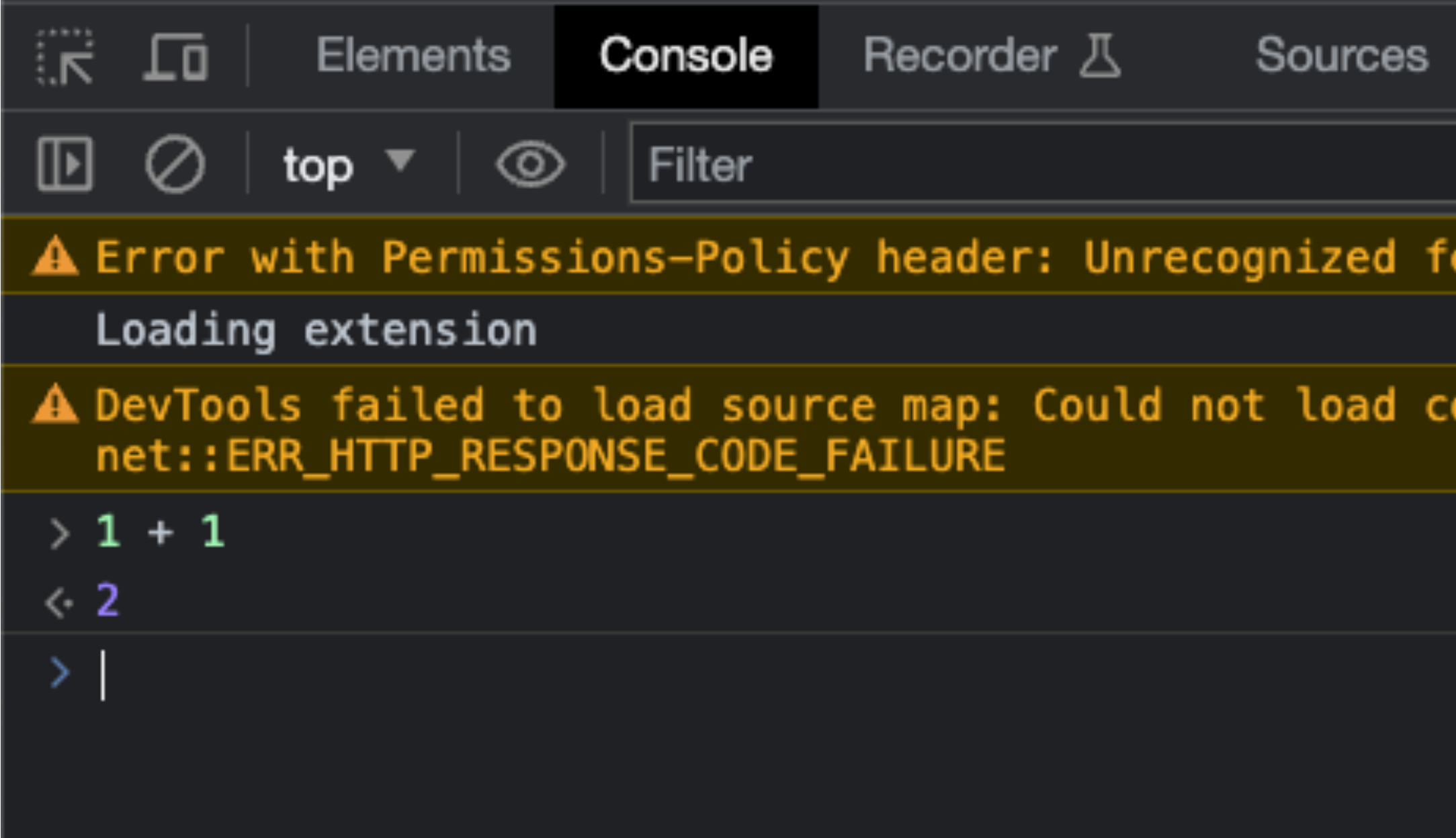
The screenshot shows the Chrome DevTools Console. The 'Console' tab is active. At the top, there are navigation icons and a filter box. Below the navigation, there are two error messages: 'Error with Permissions-Policy header: Unrecognized f...' and 'DevTools failed to load source map: Could not load c... net::ERR_HTTP_RESPONSE_CODE_FAILURE'. Below the errors, there is a JavaScript expression '> 1 + 1' which has been evaluated to '< 2'. The prompt '>' is followed by a vertical bar '|', indicating the next input.

Technical

1. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Expressions_and_Operators

Calculation in Javascript

- Expressions include:
 - Mathematical expressions;
 - Logical Expressions; as well as
 - Function call returns
- Expressions themselves are NOT full statements, they form part of the statement such as assignments



The screenshot shows the Chrome DevTools Console with the 'Console' tab selected. The top navigation bar includes 'Elements', 'Console', 'Recorder', and 'Sources'. Below the navigation bar, there are icons for 'top', a dropdown menu, and a 'Filter' input field. The console displays several messages: a yellow warning icon followed by the text 'Error with Permissions-Policy header: Unrecognized fo', a grey message 'Loading extension', and another yellow warning icon followed by 'DevTools failed to load source map: Could not load c net::ERR_HTTP_RESPONSE_CODE_FAILURE'. Below these messages, the console shows a prompt '>' followed by the input '1 + 1', which is followed by the output '<' and '2'. A new prompt '>' with a cursor is visible at the bottom.

Technical

Calculation in Javascript

```
x = 10;  
y = x * x + 2 * x + 1;
```

- These two are full statements. They must end with semicolon.
- x and y here are variables.
- When **executed** (by pressing enter), the **expressions** are **evaluated** (calculated), and their results **assigned** to Variables x and y
- If variables to be assigned doesn't exist, they will first be created (**declared**) in memory
- To checkout the values of already declared variables, type the name of the variable in the console, then press enter

Calculation in Javascript

- Write a new function in the console like this:

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- Then, execute it by calling this function:

```
circumference_circle(10);
```


Calculation in Javascript

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- This is a variable, which is declared as a **function** that can be called

Calculation in Javascript

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- This is a function declaration
- A function declaration has 2 parts

Calculation in Javascript

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- This is a function declaration
- A function declaration has 2 parts
 1. Argument list: this will be the part within the parenthesis. These **arguments** are used as **variables** inside the function, with values given during function calls.

Calculation in Javascript

```
say_hello = function(my_name, your_name) {  
    console.log("Hello", your_name, "from", my_name);  
};
```

- This is a function declaration
- A function declaration has 2 parts
 1. Argument list: a function can have multiple arguments, separated by comma

Calculation in Javascript

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- This is a function declaration
- A function declaration has 2 parts
 1. Name: you can write standard Javascript statements here
These code will be executed whenever the function is called

Calculation in Javascript

```
circumference_circle = function(radius) {  
    result = radius * 2 * 3.1415926;  
    console.log("With radius", radius);  
    console.log("The circumference is", result);  
};
```

- This is a function declaration
- A function declaration has 2 parts
 1. Subroutine: subroutine starts and ends with curly brackets
Inside, you must write lines of **full statements**

console.log()

```
console.log("With radius", radius);
```

- This is also a function
- The arguments of this function call will be **printed** (outputted) in the console
- "With radius" is a string value, it gets printed as is
- radius is a variable, it's internal value will be printed
- console.log() can have as many arguments as you want

```
console.log()
```

- Regular users usually won't access the console
- This is usually used for development and debugging
- That being said... don't write your credit card number in your console, nor in your code, nor anywhere others can see

Exercise 1

- Create `script.js` with the following inside
 - Declare a function called `calc_tax`, that takes `price` as input
 - Here's what you should output using `console.log` in 3 lines:
 - Original Price: `price`
 - GST is 7% of the `price`
 - PST is 5% of the `price`
 - Total cost is `price` plus GST and PST
- Include `script.js` in an `index.html`, open `index.html` in your browser, call the following in console:
`calc_tax(100);`

```
> calc_tax(100);  
Price: CAD 200  
GST:   CAD 14  
PST:   CAD 10  
Total: CAD 224  
>
```

Exercise 2

- At the end of your `script.js`'s function, add this:

```
return price + gst + pst;
```

```
...  
return price + gst + pst;  
}
```

- Open `index.html` in your browser, call the following in console:

```
x = calc_tax(100);
```

- Now check the value of `x`, what do you get?
- This statement is called `return`, it allows a function call to be used in an **expression** to provide values.
- The default return value is `undefined`, which doesn't mean anything

Javascript Data Types

- We have now encountered 3 data types
 - Number: integers and float numbers, e.g. 100, 3.14
 - String: text values, e.g. "Hello World!"
 - Functions: that can be called, e.g. `console.log`, `alert`, etc.
- Javascript **Variables** carry data **values** of various **types**, typing is **dynamic**
 - **Dynamic**: the same variable can take on values of all types on the fly (C and C++ doesn't support this, Python and Javascript does)